A hybrid adaptive large neighborhood search heuristic for lot-sizing with setup times

Muller, Laurent Flindt; Spoorendonk, Simon; Pisinger, David

Published in:
European Journal of Operational Research

Link to article, DOI:
10.1016/j.ejor.2011.11.036

Publication date:
2012

A hybrid adaptive large neighborhood search heuristic for lot-sizing with setup times

Laurent Flindt Muller, Simon Spoorendonk, David Pisinger*

Department of Management Engineering, Technical University of Denmark, Produktionstorvet, Building 426, DK-2800 Kgs. Lyngby, Denmark

Abstract

This paper presents a hybrid of a general heuristic framework and a general purpose mixed-integer programming (MIP) solver. The framework is based on local search and an adaptive procedure which chooses between a set of large neighborhoods to be searched. A mixed integer programming solver and its built-in feasibility heuristics is used to search a neighborhood for improving solutions. The general reoptimization approach used for repairing solutions is specifically suited for combinatorial problems where it may be hard to otherwise design suitable repair neighborhoods. The hybrid heuristic framework is applied to the multi-item capacitated lot sizing problem with setup times, where experiments have been conducted on a series of instances from the literature and a newly generated extension of these. On average the presented heuristic outperforms the best heuristics from the literature, and the upper bounds found by the commercial MIP solver ILOG CPLEX using state-of-the-art MIP formulations. Furthermore, we improve the best known solutions on 60 out of 100 and improve the lower bound on all 100 instances from the literature.

Keywords: Production, Manufacturing, Heuristics, Large scale optimization

*Corresponding author

Email addresses: lafm@man.dtu.dk (Laurent Flindt Muller), spoo@man.dtu.dk (Simon Spoorendonk), pisinger@man.dtu.dk (David Pisinger)

1. Introduction

The adaptive large neighborhood search (ALNS) heuristic is a concept introduced by Røpke & Pisinger (2006). The ALNS heuristic is a large neighborhood improvement heuristic that operates on top of a construction heuristic. The improvement is done using a local search method, e.g., simulated annealing or tabu search, choosing between different neighborhoods. In each iteration of the search a *destroy neighborhood* is chosen to destroy the current solution, and a *repair neighborhood* is chosen to repair the solution. The neighborhoods are weighted according to their success and weights are adjusted as the ALNS heuristic progresses. Destroy and repair neighborhoods are normally assumed to be searched by fast heuristics.

The main motivation for extending the ALNS heuristic to a hybrid version is that not all problem types are equally well suited for defining neighborhoods. Especially the construction and the exploration of repair neighborhoods can be a challenge, both with respect to finding a meaningful repair operation and testing feasibility of such an operation. To address this problem, we propose to use a mixed integer programming (MIP) solver in the repair phase of the ALNS heuristic. The idea is to solve a restricted subproblem that is based on a partial solution where variables are fixed (or bounded). The process of constructing a subproblem, and the following reoptimization of the subproblem with the use of a MIP solver, can in the context of an ALNS heuristic be seen as the application of a destroy and a repair neighborhood. As such the hybrid ALNS can be viewed as a specialization of the ALNS framework which simplifies the task of defining repair neighborhoods.

The reoptimization of the subproblems done in the repair neighborhoods relies heavily on primal heuristics in the MIP solver to produce good incumbent solutions since it may be too cumbersome to solve the subproblem to optimality. Heuristics found in modern MIP solvers include the local branching heuristic by Fischetti & Lodi (2003), the feasibility pump introduced by Fischetti et al. (2005) and refined by Bertacco et al. (2007); Achterberg & Berthold (2007), and the relaxation induced neighborhood search by Danna et al. (2005). Naturally such MIP heuristics are constructed in such a way that they can be applied directly to a problem without taking into account special characteristics. Also, the MIP heuristics are limited in the sense that they are only applied within the branch-and-bound tree and they are induced from the current fractional solution. The hybrid ALNS works with

different neighborhoods, outside the scope of a branch-and-bound tree, and takes historical information into account.

The ALNS framework introduced by Røpke & Pisinger (2006) has grown out of the large neighborhood search framework by Shaw (1998). The heuristic has several similarities with variable neighborhood search, see e.g., Mladenović & Hansen (1997), and hyper-heuristics, see e.g., Burke et al. (2003). However, there is no adaptiveness built into the basic idea of the variable neighborhood search. This approach mainly relies on the diversity of the neighborhoods being used. The hyper-heuristic approach operates on simpler low-level heuristics whereas the ALNS heuristic operates on neighborhoods. Furthermore, to make the search adaptive, an evaluation function is used to calculate a score for each low-level heuristic. This score is used in a roulette-wheel selection mechanism to choose a neighborhood for the next iteration. ALNS heuristics have been implemented for vehicle routing problems with great success, see Røpke & Pisinger (2006); Pisinger & Røpke (2007). Examples of an application of the framework outside a routing problem context are few: Cordeau et al. (2010) schedule technicians and tasks in a telecommunications company and Muller (2009) presents an ALNS heuristic for the resource-constrained project scheduling problem. For a recent survey on large neighborhood search and the ALNS framework we refer to Pisinger & Røpke (2010).

The lot sizing problem (LSP) with setup times and setup costs can be defined as follows: Given one resource, schedule the production of a set of items over a given number of time periods such that all demands of items are met, and such that the capacity of the resource is not exceeded. The production of an item and each setup of production consumes capacity on the resource and has a cost. The difference between setup times and setup costs, is that setup times consume an amount of capacity on the resource, while setup costs are an extra cost incurred in the objective function. The LSP with setup times and setup costs is $\mathcal{NP}$-hard, see e.g., Pochet & Wolsey (2006). Maes et al. (1991) show that the problem remains $\mathcal{NP}$-hard when no setup costs are present (in fact just finding a feasible solution is $\mathcal{NP}$-hard). Heuristics for the LSP with setup times and setup costs include the Lagrangian relaxation based heuristic of Trigeiro et al. (1989), the variable neighborhood search heuristic of Hindi et al. (2003), and the cross entropy-Lagrangian hybrid heuristic of Caserta & Rico (2009). Gopalakrishnan et al. (2001) present a tabu search heuristic for a variant of the LSP with setup times and setup costs, where setups can carry over from one time period to

the next.

In the multi-period carry-over variant Sahling et al. (2009) present a fix-and-optimize heuristic that repeatedly solves a series of subproblems with a MIP solver. The approach is somewhat similar to the hybrid method presented in this paper although it has no adaptiveness or randomness built-in, instead Sahling et al. (2009) consider a large number of predetermined subproblems and optimize over all of them. For some comparisons see for instance Jans & Degraeve (2007) or Buschkühl et al. (2010). Exact approaches for the LSP with setup times and setup costs include branch-and-cut algorithms by Belvaux & Wolsey (2000); Wolsey (2002); Miller et al. (2000) and a branch-and-price algorithm by Degraeve & Jans (2007). The good performance of the branch-and-cut algorithms suggest that using a MIP solver to solve restricted subproblems of the LSP with setup times and setup costs can be done in reasonable time.

A recent study by Sural et al. (2009) shows that the standard benchmark instances of Trigeiro et al. (1989) are considerably harder when setup costs are removed. Furthermore, Sural et al. (2009) consider an extension of the standard (heterogeneous) instances denoted the homogeneous instances where all holding costs are equal. Their experiments showed that the homogeneous instances have even larger integrality gaps than the heterogeneous instances. It is thus of interest to develop heuristics for the case with setup times and no setup costs, as considered in the following. Moreover, the problem appears frequently in the industry, where different tools are used to produce the items: The tools are a one-off investment so changes of tools only involve a setup time. Papers relating to the LSP with setup times and no setup costs include the MIP based heuristic of Denizel & Süral (2006), and the Lagrangian heuristic of Sural et al. (2009). In the following we will refer to the LSP with setup times and no setup costs as the LSPST.

The contribution of this paper is an ALNS heuristic which combines the speed and flexibility of modern MIP solvers with the diversity of the ALNS heuristic, creating a "hybrid" approach. The repair neighborhoods employ the MIP solver in a generic fashion and neighborhoods are thus applicable to a large variety of problems. An evaluation of the hybrid ALNS heuristic is applied to the LSPST on a set of instances found in the literature. The ALNS heuristic outperforms ILOG CPLEX (applied to two state-of-the-art MIP formulations) and the current best heuristic of Sural et al. (2009) both with respect to the quality of solutions and lower bounds. During the experiments we found 60 new best upper bounds (for the 100 instances also considered

by Sural et al. (2009)), and improved all lower bounds. This indicates the usefulness of the hybrid ALNS approach.

The paper is organized as follows: Section 2 gives an outline of the ALNS framework and describes the hybrid variant proposed in this paper, Section 3 presents an application of the hybrid ALNS heuristic to the LSPST, and Section 4 contains the experimental results performed on the instances of Sural et al. (2009) which is an extension of the instances of Trigeiro et al. (1989), and on a new set of larger instances. Section 5 concludes the paper and suggests new directions for future research.

2. A hybrid ALNS heuristic

We begin with an outline of the ALNS framework as described by Pisinger & Røpke (2007) for a combinatorial problem $\min\{f(x) \mid x \in X\}$. The framework is divided into three parts, i) a master local search framework, ii) a set of large neighborhoods that either destroy a solution or repair a partial solution, and iii) a procedure for choosing neighborhoods which adapts to the considered instance based on past performance. Following this, we present the hybrid ALNS algorithm.

At the top level (also denoted *master level*) any local search heuristic can be applied, e.g., simulated annealing, tabu search, guided local search, or GRASP (greedy randomized adaptive search procedure). A neighborhood of a solution is a set of solutions obtained by performing some operation on the original solution. In large neighborhoods these operations involve changing several settings in the solution at once, leading to a neighborhood of potentially exponential size. Roulette wheel selection is used for choosing neighborhoods, where the weight of a neighborhood is based on historical success. Hence, successful neighborhoods have a higher probability to be used as time passes, although all neighborhoods have a small chance of being chosen to ensure diversity.

The ALNS framework can be described as follows: Given a starting solution, the heuristic iteratively tries to improve it by exploring various neighborhoods. Each neighborhood operates on a set of elements, e.g. variables in a MIP-model, customers in a transportation problem, or items in a lot-sizing problem. The set of neighborhoods is divided into destroy neighborhoods N^- and repair neighborhoods N^+. Given a current solution x a destroy neighborhood $n^- \in N^-$ performs an operation on x, stores the removed elements in an *item bank* B and leaves a partial solution $\overline{x}$. A repair neighborhood

inserts elements from the item bank into $\overline{x}$ creating a new solution x'. In the case of the hybrid ALNS heuristic presented in this paper, the repair neighborhoods make use of a MIP solver. A roulette wheel for each of the sets N^- and N^+ is used in each iteration to choose which destroy and which repair neighborhood should be used. This is based on a weight π for each neighborhood that is initialized at the beginning according to the quality of the neighborhood (this is a user defined consideration to be made a priori). During the local search the weights are updated according to the quality of the solutions produced with the given neighborhoods. The motivation behind this is, that not all neighborhoods perform equally well on all problem instances – hence the weights of the neighborhoods adapt to the instance during the execution of the algorithm and hopefully produce better solutions overall.

As mentioned above, the weights of the neighborhoods are updated according to how successful a neighborhood is in obtaining better and new solutions. In the paper by Pisinger & Røpke (2007) the weights are updated as follows: the course of the algorithm is divided into time segments (e.g. number of iterations). In each time segment t, two scores are maintained for each neighborhood i: an *observed* weight $\overline{\pi}_{i,t}$ records the actual performance of neighborhood i in each iteration of the segment, while a *smoothened* weight $\pi_{i,t}$ is calculated at the end of the segment on the basis of $\overline{\pi}_{i,t}$, the number of times $a_{i,t}$ neighborhood i has been chosen in time segment t, and previous values of $\pi_{i,t}$. It is the smoothed weight which is used in the roulette wheel for the subsequent time segment. A *reaction factor* r controls how much the roulette weight depends on score in the most recent time segment t. The smoothed weight is updated as follows:

$$\pi_{i,t+1} = r\frac{\overline{\pi}_{i,t}}{a_{i,t}} + (1-r)\pi_{i,t}.$$

A low reaction factor keeps the weight at about the same level during the algorithm. A neighborhood i has the probability:

$$p_t(i) = \frac{\pi_{i,t}}{\sum_{j \in N} \pi_{j,t}}$$

of being chosen in time segment t.

In Figure 1, the pseudo-code is given for the ALNS framework. The criteria for accepting a new solution in line 5 depends on the choice of local

```
ALNS
 1  x is an initial solution; set x* := x
 2  repeat
 3           Choose n⁻ ∈ N⁻ and a n⁺ ∈ N⁺ based on π
 4           Generate solution x′ based on x, n⁻, and n⁺
 5           if x′ is accepted
 6              then x := x′
 7           if x′ < x*
 8              then x* := x′
 9           Update π for N⁻ and N⁺
10     until stop criterion is met
11  return x*
```

Figure 1: Pseudo-code overview of the ALNS framework.

search framework and the score update on line 9 can be performed using different strategies. The choices made for the hybrid ALNS heuristic will be described in the following sections.

The basic idea behind the proposed hybrid ALNS heuristic is, that instead of designing special purpose repair neighborhoods, which may not always be straight forward, we use a MIP solver in order to repair (or reoptimize) a solution. Each destroy neighborhood selects a number of variables from the MIP model based on the current solution. These variables are "free" in the sense that no additional constraints are imposed on them in the subproblem constructed by the chosen repair neighborhood. Depending on the repair neighborhood the remaining variables of the subproblem are either fixed or bounded based on their values in the current solution. Thus the MIP based repair neighborhood will consequently search a neighborhood around the current solution. Using a MIP solver as a repair neighborhood provides an easy tool to calculate lower bounds during the search. We propose that, when an improved solution is found, the root node of the MIP is resolved with all variables freed, and the current solution as an initial upper bound. For modern MIP solvers an initial upper bound is used for both pre-processing and reduced cost fixing during the optimization. Hence, a good initial upper bound may yield improved lower bounds compared to solving the root node with no (or a bad) initial solution. This way, the ALNS heuristic can provide

valid lower bounds and an estimation of the solution quality based on the integrality gap.

3. An application of ALNS to the LSPST

In this section we present an application of the hybrid ALNS heuristic to the LSPST. First, a description of the "standard" mathematical model is given, followed by a description of a transportation reformulation. We then turn our attention to the hybrid ALNS heuristic and present the master level local search procedure, followed by a description of an adjusted weight calculation method used in the adaptive procedure. A description of the neighborhoods employed is given, and finally the parameter values are presented.

3.1. Problem description

Standard formulation. This section briefly presents the "standard" mathematical formulation (see e.g., Belvaux & Wolsey (2000)) of the LSPST. Let $I = \{1, \ldots, n\}$ be the set of items and $T = \{1, \ldots, m\}$ be the set of time periods. The data set is given as follows: $h^i \geq 0$ is the unit inventory cost of item i, $d_t^i \geq 0$ is the demand of item i at time t, $\alpha_t^i \geq 0$ is the capacity used for producing item i at time t, $\beta_t^i \geq 0$ is the capacity used for setting up the production of item i at time t, $C_t \geq 0$ is the capacity of the resource at time t, and M is a sufficiently large constant. The variables are give as follows: s_t^i is the number of units of item i in stock at the end of time t, x_t^i is the number of units of production of item i at time t, and y_t^i indicates if a setup for production of item i at time t has been done. The y-variables are binary, while the remaining variables are positive and continuous. The standard mathematical (STD) formulation for the LSPST is:

$$\min \quad \sum_{i \in I} \left(\sum_{t \in T} h^i s_t^i \right) \tag{1}$$

$$\text{s.t.} \quad s_{t-1}^i + x_t^i = d_t^i + s_t^i \qquad t \in T,\ i \in I \tag{2}$$

$$x_t^i \leq M y_t^i \qquad t \in T,\ i \in I \tag{3}$$

$$\sum_{i \in I} \left(\alpha_t^i x_t^i + \beta_t^i y_t^i \right) \leq C_t \qquad t \in T \tag{4}$$

$$s_t^i, x_t^i \geq 0,\ y_t^i \in \{0, 1\} \qquad t \in T,\ i \in I \tag{5}$$

The objective (1) is to minimize the overall holding cost. Constraints (2) ensure flow conservation of each item. That is, items in stock plus the items produced in a time period must equal the number of items demanded in this time period plus the number of items in stock after this time period. Constraints (3) ensure that production of an item can only occur if the resource is set up to produce that item. Constraints (4) guarantee that the production and setup capacity usages cannot exceed the resource capacity. Finally, the domains of the variables are specified by constraints (5).

Transportation reformulation. In the paper by Denizel & Süral (2006) three different strong reformulations of LSPST are examined. One of these, the transportation problem reformulation, seems to perform the best and is also the formulation employed for the heuristic presented by Sural et al. (2009) (currently the best heuristic for the problem). We will examine both the standard formulation and the strong transportation reformulation when comparing the hybrid ALNS heuristic with ILOG CPLEX. Let $z^i_{tr} \geq 0$ be the quantity produced in period $t \in T$ to satisfy the demand of item $i \in I$ in period $r \in T$, where $r \geq t$. The remaining variables are as for the standard model. The transportation (TP) reformulation can be written as:

$$\min \sum_{i \in I} \left(\sum_{r \in T} \sum_{t=1}^{r-1} (r-t) h^i z^i_{tr} \right) \tag{6}$$

$$\text{s.t.} \sum_{t=1}^{r} z^i_{tr} = d_{ir} \qquad r \in T,\ i \in I \tag{7}$$

$$z^i_{tr} \leq d_{ir} y^i_t \qquad t \in T,\ r = t, \ldots, m,\ i \in I \tag{8}$$

$$\sum_{i \in I} \left(\sum_{r=t}^{m} \alpha^i_t z^i_{tr} + \beta^i_t y^i_t \right) \leq C_t \qquad t \in T \tag{9}$$

$$y^i_t \in \{0, 1\} \qquad t \in T,\ i \in I \tag{10}$$

$$z^i_{tr} \geq 0 \qquad t \in T,\ r = t, \ldots, m,\ i \in I \tag{11}$$

The objective (6) is again to minimize the overall holding cost. Constraints (7) ensure that the total production of item i in periods 1 through r is equal to the demand in period r. Constraints (8) ensure that production of an item can only occur if the resource is set up to produce that item. Constraints (9) guarantee that the production and setup capacity usages cannot exceed the

resource capacity. The domains of the variables are specified by constraints (10) and (11).

Although Denizel & Süral (2006) showed that the TP model is easier to solve than the STD model, we are going to solve the LSPST heuristically with several variables fixed in ALNS. Preliminary computational experiments with the two MIP formulations indicate that the STD formulation is preferable with respect to reoptimization times compared to the TP formulation. This may be due to the quadratically increase in variables and constraints for the TP formulation when the number of time periods increases. Also, it appears that the primal heuristics of ILOG CPLEX are more efficient for the STD formulation than for the TP formulation. Therefore, we have decide to base the hybrid ALNS algorithm on the STD formulation.

3.2. Local search

In this paper, steepest descent has been chosen as the master level local search procedure in ALNS. Since the destroy neighborhoods merely free a subset of the variables, and because the MIP repair neighborhoods use the current solution as an initial upper bound, it is always possible for the MIP solver to find that solution again. Therefore, the MIP solver never returns a solution that is worse than the current one. Hence, a selection process for choosing worse solutions would not be relevant.

To diversify the search, the algorithm is restarted at different solutions when no improvements have occurred in a number of iterations (chosen to be equal to the segment size for updating the neighborhood weights). For the initial restart the second best solution is chosen (the current solution is the best solution). In subsequent restarts, the local search either switches back to the best solution if it is not the current one or switches to the next best solution that has not previously been used for a restart. The reason for returning to the best solution in an attempt to find further improvements is that the neighborhoods may have obtained different scores in the meantime yielding a diversified exploration of the neighborhood of that solution.

To speed up the subproblem solution process, we suggest to limit the number of explored branch nodes or terminate the search after a given time limit. MIP heuristics are applied in the MIP solver to obtain feasible solutions rapidly.

3.3. Adaptive weight adjustments

In this paper we employ the same scoring scheme as (Røpke & Pisinger, 2006; Pisinger & Røpke, 2007). That is, if a neighborhood i produces a new best incumbent solution it is awared a score of k_0, i.e., the observed weight is updated as $\bar{\pi}_{i,t} = \bar{\pi}_{i,t} + k_0$, and if it produces a local improvement it is awarded a score of k_1.

3.4. Destroy neighborhoods

Except for the random removal neighborhood each neighborhood focuses on specific structural disadvantages in a solution to the LSPST, e.g., too many items in stock, or a frequent change of the production of items. A parameter Q controls how large a part of the current solution is destroyed. It is measured as a percentage of the combined production of the LSPST. The destroy neighborhoods are divided in two steps: i) a removal candidate set, R, of sets of production variables, i.e., sets of x-variables, is constructed based on the chosen destroy neighborhood, ii) iteratively a set F of production variables is constructed by selecting sets of variables from R until their cumulative production values correspond to Q, or no more sets of variables remain. The variables of the set F are the variables that will be freed in the subsequent reoptimization. Depending on the destroy neighborhood chosen, the sets of variables are either randomly selected or randomly selected based on some weight. In the following let $(\bar{x}, \bar{y}, \bar{s})$ denote the current solution. There are in total six destroy neighborhoods:

Random. Frees production variables at random throughout the production plan:

$$R = \left\{ \{x_t^i\} : \bar{x}_t^i > 0, t \in T, i \in I \right\}.$$

The set F is constructed by choosing sets of variables from R randomly. The neighborhood is good at diversifying the search, and hence it is useful if the search is stuck in a local minimum.

Production causing stock. Frees production variables that cause items to be placed in stock. Hopefully, the production can be inserted at a later time in the production plan, saving inventory expenses:

$$R = \left\{ \{x_t^i\} : \bar{s}_t^i > 0 \wedge \bar{s}_{t+1}^i > 0, t \in T, i \in I \right\}.$$

The set F is constructed by choosing sets of variables from R randomly.

Capacity critical. Frees production of items in time steps, where some resource is fully loaded. This allows for a reshuffling of the production:

$$R = \left\{ \{x_t^i\} : \bar{x}_{t^*}^i > 0 \wedge (t = t^* \vee t = t^* - 1 \vee t = t^* + 1), i \in I \right\},$$

where $t^* = \arg\max\{\sum_I \bar{x}_t^i : t \in T\}$, i.e., t^* is the time period with the most combined production. Variables for time steps immediately preceding and succeeding t^* are included to open up for the possibility of shifting the production between these time periods. The set F is constructed by choosing sets of variables from R randomly.

Stocked items. Frees production of items that have the largest amount of units in stock throughout the production plan. The idea is to attempt a reshuffle of stocked items between those types of items that are favorable to put in stock, e.g., due to low holding cost:

$$R = \left\{ \{x_t^i : t \in T\} : \sum_{t \in T} \bar{s}_t^i > 0, i \in I \right\}$$

The set F is constructed by choosing sets of variables from R randomly by roulette wheel selection, where each set $S \in R$ corresponding to some item i^*, has weight $\sum_{t \in T} \bar{s}_t^{i^*}$. At least two item types are freed.

Time periods with high stock density. Frees production from time periods where many items are in stock. The idea is to reshuffle the production (and thereby the stocked items) into the previous or succeeding time periods. The time periods are sorted according to the number of stocked items. When the production in a time period t is cleared, the time periods immediately preceding and succeeding t are also cleared:

$$R = \left\{ \{x_{t-1}^i, x_t^i, x_{t+1}^i : i \in I\} : t \in T \right\}$$

The set F is constructed by choosing sets of variables from R randomly by roulette wheel selection, where each set $S \in R$ corresponding to some triple of time $(t^* - 1, t^*, t^* + 1)$ has weight $\sum_{i \in I}(\bar{s}_{t^*-1}^i + \bar{s}_{t^*}^i + \bar{s}_{t^*+1}^i)$.

Production higher than demand. When the production is high compared to the demand of the corresponding item in a given time period t, several items are put in stock. Often a solution can be shifted, such that

the majority of the production is moved to the following time period, so we define:

$$R = \left\{ \{x^i_{t-1}, x^i_t, x^i_{t+1}\} : t \in T, i \in I \right\}$$

The set F is constructed by choosing sets of variables from R randomly by roulette wheel selection, where each set $S \in R$ corresponding to some item i^* and some triple of time $(t^* - 1, t^*, t^* + 1)$ has weight $\bar{x}^{i^*}_{t^*} - d^{i^*}_{t^*}$.

3.5. Repair neighborhoods

Let F be the set constructed by the application of one of the destroy neighborhoods just described. In the following let $\bar{F}$ denote the set production variables not in F, i.e., the set of variables which are not freed. The two MIP repair neighborhoods employed for the LSPST are:

Bound by solution value. Bound variables $x \in \bar{F}$ to a fraction of their current value, i.e., if x^i_t is a variable in $\bar{F}$, fix $x^i_t \geq \delta \bar{x}^i_t$ for some value of δ. An appropriate value of δ was chosen as $\delta = 0.5$ for the studied problem. Production variables x are linked to the setup variables y. Hence, whenever $\bar{x}^i_t > 0$ we bound the variable $y^t_t = 1$. Stock is generally to be avoided, therefore we avoid to fix any of the stock variables s from below so that we do not accidentally force unnecessary stock. When employing this repair neighborhood the optimization problem (1) – (5) is extended with the following constraint

$$x^i_t \geq \delta \bar{x}^i_t, \quad \forall x^i_t \in \bar{F}$$

Fix by solution value. Fix variables $x \in \bar{F}$, to their value in the current solution, where the production equals the demand, i.e., we fix $x^i_t = \bar{x}^i_t$ whenever we have $\bar{x}^i_t = d^i_t$. In solutions for the LSPST, this scenario happens frequently in consecutive time periods for the production of an item. In order to allow some diversification in the production we only fix production when there is no stock in the preceding and the succeeding time periods, i.e., for a variable x^i_t it must hold that $s^i_{t-1} = s^i_{t+1} = 0$. Let $\bar{\bar{F}} \subseteq \bar{F}$ be the subset of variables of $\bar{F}$ for which this condition holds. When employing this repair neighborhood the optimization problem (1) – (5) is extended with the following constraint

$$x^i_t = \bar{x}^i_t, \quad \forall x^i_t \in \bar{\bar{F}}$$

3.6. ALNS parameters

At the master level, a time limit of 300 seconds is chosen as the termination criteria of the steepest descent local search. The reaction factor r is set fairly high at 0.8 since we expect few iterations, and therefore would like the neighborhood weights to converge fast. All neighborhoods are initialized with a weight of 100. Through experimentation, the values 20 and 13 where chosen for k_0 and k_1 respectively when adaptively adjusting the weights of the neighborhoods. During the first iteration, an estimate of the overall number of iterations is calculated based on the running time of that iteration, i.e., the $max\_iterations = time\_limit/time\_first\_iteration$. The segment size for updating the weights is set to one hundredth of the estimated number of overall iterations, or at least 10 and at most 50 iterations. As mentioned earlier the parameter Q controls how large a part of the current solution is destroyed. Muller (2009) suggests an exponential decrease of Q in the number of iterations. In this paper we propose a linear decrease beginning at $Q_{start} = 0.4$ and decreasing towards $Q_{end} = 0.1$. Since the number of iterations is unknown we calculate the decrease based on the maximum running time, i.e, $Q = Q_{start} - (Q_{start} - Q_{end}) \cdot (time\_current/time\_limit)$. The linear decrease provides room for large neighborhoods in more iterations which is crucial when few iterations are explored.

3.7. Overview

Figure 2 shows the pseudo-code for the complete algorithm. The initial solution is found by solving the root node of the branch-and-bound-tree and returning the best heuristic solution found by the MIP solver. For all the considered test instances this produced a feasible initial solution. The stop criteria used in the call to the MIP solver in line 6 is to solve the root node of the branch-and-bound tree and then return the best heuristic solution found. For this call the MIP solver is initialized with the current solution.

4. Experimental results

In this section, we compare the ALNS heuristic to ILOG CPLEX with default settings (using both the STD and the TP formulations) and to the heuristic of Sural et al. (2009), which is currently the best for the problem considered. The experiments are performed on a 2.66 GHz Intel(R) Xeon(R) X5355 machine with 8 GB memory using ILOG CPLEX version 12.1. For the result reported by Sural et al. (2009) an IBM PC with an Intel Pentium

```
HYBRID-ALNS
 1  x is an initial solution; set x* := x
 2  repeat
 3          Choose n⁻ ∈ N⁻ and a n⁺ ∈ N⁺ based on π
 4          Construct the set F (and R) based on n⁻ and x.
 5          Construct a restricted MIP model based on
            F̄, n⁺ and x
 6          Solve problem with a MIP solver (subject to
            stopping criterion)
 7          Collect all solutions found by MIP solver into a pool
 8          if no solution better than x found for t_noimprovment
 9            then if x = x*
10                    then x := x*
11                    else Set x to second best solution not
                           previously used for restarting
12            else Let x' be the best found solution.
13                 if x' is better than x
14                   then x := x'
15                 if x' better than x*
16                   then x* := x'
17                        Use MIP solver to resolve problem
                          with x* as input obtain LB
18          Update π for N⁻ and N⁺
19     until stop criterion is met
20  return x*
```

Figure 2: Pseudo-code overview of the ALNS framework.

IV processor was employed, which is about 3 times slower than the processor used for these tests according to SPEC (`www.spec.org`). The time limit for the ALNS heuristic and for ILOG CPLEX is 300 seconds, which is also one of the stopping criteria used by Sural et al. (2009). For the ALNS heuristic results are calculated as the average of 10 runs.

4.1. Instances

The considered test instances are the same as those used by Sural et al. (2009). These instances have been generated on the basis of the instances of Trigeiro et al. (1989) by setting the setup costs to 0, and setting all zero demand to 2. The base set of Trigeiro et al. (1989) are divided into four groups: the five instances `G51`–`G55` each have 12 items and 15 time periods, the five instances `G56`–`G60` each have 24 items and 15 time periods, the five instances `G66`–`G70` each have 12 items and 30 time periods, and the five instances `G71-G75` each have 24 items and 30 time periods. In addition to these instances, Sural et al. (2009) generate four new groups having 10 time periods and two new groups having 15 time periods by taking the original instances and reducing the number of time periods to respectively 10 and 15. These are in the following denoted by appending `-10` and `-15` to the name of the original instance. This results in 50 heterogeneous instances. Additional 50 homogeneous instances were created by Sural et al. (2009) on the basis of the heterogeneous instances by setting all holding costs to 1.

In order to experiment with larger (and harder) instances, we have generated a number of new instances in a similar way as Sural et al. (2009): For each of the original instances containing 15 time periods an instance containing respectively 30 and 45 time periods is created by concatenating the 15 time period instance (two respectively three times). Likewise, for each of the original instances containing 30 time periods, an instance containing respectively 60 and 90 time periods is created by concatenation. These are in the following denoted by appending `-30`, `-45`, `-60` and `-90` to the name of the original instance. Again, a further set of homogeneous instances is created on the basis of these by setting all holding costs to 1. This results in a total of 40 new heterogeneous and 40 new homogeneous instances. The total number of instances considered is thus 180.

In the following experiments, the class S refers to the instances by Sural et al. (2009) and the class M refers to the instances generated in this paper.

	ALNS				MIP STD			MIP TP			Sural et al. (SDW)		
Group	LB	UB	UB*	gap	LB	UB	gap	LB	UB	gap	LB	UB	gap
S homo	13.42	**0.18**	0.01	16.98	7.17	0.30	19.36	**5.00**	0.46	**6.28**	18.46	0.63	27.78
S hetero	11.80	**0.13**	0.02	23.71	4.67	0.19	14.26	**2.35**	**0.13**	**2.83**	15.71	2.98	25.51
M homo	15.44	**0.67**	0.07	**20.40**	20.56	1.30	161.75	**14.73**	1.89	20.99	-	-	-
M hetero	11.68	**1.11**	0.28	**15.55**	16.92	1.19	140.80	**11.03**	1.66	15.63	-	-	-

Table 1: Comparison of the ALNS heuristic with the STD and TP formulations solved with default ILOG CPLEX settings and the SDW heuristic of Sural et al. (2009). The LB column is the average deviation in percent of the lower bound from the best upper bound found across all algorithms calculated as $UB - LB/UB$, thus smaller is better. The UB column is likewise the average deviation of the upper bound from the best found solution, the UB* column is the best solution found across the 10 runs of the ALNS heuristic. The gap column is the average integrality gap in percent at the point where the procedure stops using the lower and the upper bounds calculated by that procedure. For the ALNS heuristics the results are reported as average of 10 runs. For each line the best value across the algorithms are indicated in **boldface**.

4.2. Comparison

Table 1 and Table 2 shows a comparison, for bounds and time respectively, of the results obtained by applying the ALNS heuristic, ILOG CPLEX with default settings (using both the STD and TP formulations), and the best heuristic of Sural et al. (2009) (SDW) to the benchmark instances. We remind the reader that for the ALNS heuristic, the results are the average of 10 runs. The results shown are broken down by classes (S and M), and by heterogeneous and homogeneous instances. The time column is, for the ALNS heuristic, the average time taken to find the best solution, and for the MIP models it is the time taken to find the best solution (given the 300 second time limit). For SDW it is the total time used by the algorithm. Many of the instances (especially in class S) can be solved to optimality within the 300 second time limit when employing ILOG CPLEX. Detailed results for each instance may be found in Appendix Appendix A. Note that for the MIP models, some of the reported times may exceed the 300 second time limit due to close-down of ILOG CPLEX.

When considering the instances of Sural et al. (2009), we see from Table 1 that both the ALNS heuristic and the MIP solver outperform the best heuristic (SDW) of Sural et al. (2009) both with regards to the quality of the lower

	ALNS	MIP STD		MIP TP		Sural et al. (SDW)
Group	tb(s)	tb(s)	tt(s)	tb(s)	tt(s)	tt(s)
S homo	64.44	128.42	196.17	126.66	204.68	**8.99**
S hetero	57.96	90.51	110.34	80.67	109.00	**12.14**
M homo	**229.50**	264.18	300.08	291.74	323.52	-
M hetero	**232.53**	261.07	287.63	262.51	316.30	-

Table 2: Comparison of the ALNS heuristic with the STD and TP formulations solved with default ILOG CPLEX settings and the SDW heuristic of Sural et al. (2009). The tb(s) column is the average time used to find the best solution and the tt(s) is the total solution time by the ALNS heuristic and the MIP solver on the STD and TP formulations, and for the heuristic of Sural et al. (2009) it is the average of the total times reported in that paper. For the ALNS heuristics the results are reported as average of 10 runs. For each line the best value across the algorithms are indicated in **boldface**.

and upper bounds. Taking the best upper and lower bounds found by either the ALNS heuristic or the MIP solver we find 24 new best upper bounds and 50 new best lower bounds (out of 50) for the homogeneous instances, and 36 new best upper bounds and 50 new best lower bounds (out of 50) for the heterogeneous instances (see the next section for details). Although, the TP formulation finds equally good upper bounds on the heterogeneous class S instances, it can be seen that the ALNS heuristics consistently finds the best upper bounds. Except for the heterogeneous class S instances the STD formulation outperforms the TP formulation with regard to upper bounds. The TP formulation produces better lower bounds than both the STD formulation and the ALNS heuristic. However, when considering the M class, the computed gaps are smaller for the ALNS heuristic because it finds better upper bounds than the TP formulation. A little surprisingly the ALNS heuristic actually finds better lower bounds on the class M instances than the STD formulation although the heuristic is derived from that model.

The time comparison in Table 2 clearly shows that the heuristic of Sural et al. (2009) is by far the fastest. When comparing the ALNS heuristic and the MIP formulations, it is clear that the ALNS heuristic is much faster at finding a good solution than both of the MIP formulations. This indicates the the running time of the ALNS heuristic may be decreased from the 300 second limit and still produce good upper bounds.

We measure the effectiveness of the neighborhoods by a percentage which

is calculated based on the number of iterations a neighborhood was chosen that lead to a solution that was at least as good as the current incumbent divided by the total number of iterations that did not lead to a worse solution. For the destroy neighborhoods the neighborhoods performed equally well within a $\pm 3\%$ range and for the repair neighborhoods the range was within $\pm 2\%$ range.

5. Conclusion

We have presented a hybrid heuristic solution approach based on the ALNS framework where a MIP solver is used in the repair phase. This results in a general hybrid ALNS heuristic where i) the "difficult" part of creating efficient repair neighborhoods has been eliminated and ii) the strength of modern MIP solvers can be exploited. The framework is particularly well suited for problems where finding a feasible solution is difficult (possibly $\mathcal{NP}$-hard).

The proposed hybrid algorithm has been applied to the LSPST and the computational results indicate that the heuristic is very competitive. On two sets of benchmark instances from the literature the ALNS heuristic is, within a 300 second time limit, able to produce significantly better upper and lower bounds than previously reported. Also the ALNS heuristic was able to produce better average solution for one set and equally good solutions for the other set when compared to two formulations solved by a commercial solver. On two new sets of larger instances the ALNS heuristic outperforms the two formulations solved by a commercial solver with regard to the quality of the upper bound. Furthermore, the lower bounds produced by the ALNS heuristic is competitive with the bounds of the transportation reformulation model. Both the ALNS heuristic and ILOG CPLEX solving the STD and the TP formulations outperform the current best heuristic found in the literature, with respect to the quality of the solution and the lower bound returned. This indicates that it may be beneficial to use general MIP based repair neighborhoods in combination with problem specific destroy neighborhoods in ALNS.

Taking the best upper and lower bounds found by either the ALNS heuristic or the MIP solver we were able to improve on 24 (out of 50) upper bounds and all lower bounds for the homogeneous benchmark instances of Sural et al. (2009), and improve on 36 (out of 50) upper bounds and all lower bounds for the heterogeneous instances of Sural et al. (2009).

It is somewhat surprising that the LSPST model was more suited for the hybrid ALNS framework than the stronger TP formulation. As previously mentioned this may be due to scalability issues in the TP formulation when the number of time periods grow.

A suggestion for a future improvement is to apply the hybrid ALNS heuristic within the reoptimization process in the repair neighborhood. When solving larger problems the MIP solver may become too slow to use in the repair neighborhoods, and it may be beneficial to apply a meta-heuristic approach to reoptimize the subproblem. This approach can be applied recursively until the subproblems are small enough for the MIP solver to be handled efficiently.

Acknowledgments

Thanks to Stefan Røpke for valuable heuristic suggestions and advice on the workings of the ALNS framework.

Appendix A. Detailed results

Table A.3 and Table A.4 lists detailed results for the class S homogeneous and heterogeneous instances taken from Sural et al. (2009), and Table A.5 and Table A.6 lists detailed results for the class M heterogeneous and homogeneous instances of this paper. For the ALNS algorithm the results are averages of 10 runs.

References

Achterberg, T., & Berthold, T. (2007). Improving the feasibility pump. *Discrete Optimization*, *4*, 77–86.

Belvaux, G., & Wolsey, L. (2000). bc–prod: A specialized branch-and-cut system for lot-sizing. *Management Science*, *46*, 724–738.

Bertacco, L., Fischetti, M., & Lodi, A. (2007). A feasibility pump heuristic for general mixed-integer problems. *Discrete Optimization*, *4*, 63–76.

Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., & Schulenburg, S. (2003). Hyper-heuristics: An emerging direction in modern search technology. In F. Glover, & G. Kochenberger (Eds.), *Handbook of Meta-heuristics* chapter 16. (pp. 457–474). Kluwer.

Buschkühl, L., Sahling, F., Helber, S., & Tempelmeier, H. (2010). Dynamic capacitated lot-sizing problems: a classification and review of solution approaches. *OR spectrum*, *32*, 231–261.

Caserta, M., & Rico, E. (2009). A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times. *Computers & Operations Research*, *36*, 530–548.

Cordeau, J.-F., Laporte, G., Pasin, F., & Ropke, S. (2010). Scheduling technicians and tasks in a telecommunications company. *Journal of Scheduling*, *13*, 393–409.

Danna, E., Rothberg, E., & Pape, C. L. (2005). Exploring relaxation induced neighborhoods to improve mip solutions. *Mathematical Programming*, *A*, 71–90.

Degraeve, Z., & Jans, R. (2007). A new dantzig-wolfe reformulation and branch-and-price algorithm for the capacitated lot-sizing problem with setup times. *Operations Research*, *55*, 909–920.

Denizel, M., & Süral, H. (2006). On alternative mixed integer programming formulations and LP-based heuristics for lot-sizing with setup times. *Journal of the Operational Research Society*, *57*, 389–399.

Fischetti, M., Glover, F., & Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, *104*, 91–104.

Fischetti, M., & Lodi, A. (2003). Local branching. *Mathematical Programming*, *98*, 23–47.

Gopalakrishnan, M., Ding, K., Bourjolly, J., & Mohan, S. (2001). A tabu-search heuristic for the capacitated lot-sizing problem with set-up carry-over. *Management Science*, *47*, 851–863.

Hindi, K., Fleszar, K., & Charalambous, C. (2003). An effective heuristic for the clsp with set-up times. *Journal of the Operational Research Society*, *54*, 490 – 498.

Jans, R., & Degraeve, Z. (2007). Meta-heuristics for dynamic lot-sizing: A review and comparison of solution approaches. *European Journal of Operation Research*, *177*, 1855–1875.

Maes, J., McClain, J., & Van Wassenhove, L. (1991). Multilevel capacitated lotsizing complexity and LP-based heuristics. *European Journal of Operational Research*, *53*, 131–148.

Miller, A., Nemhauser, G., & Savelsbergh, M. (2000). *Solving multi-item capacitated lot-sizing problems with setup times by branch-and-cut*. Technical Report 39 Center for Operations Research and Econometrics, Universite Catholique de Louvain, Belgium.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, *24*, 1097–1100.

Muller, L. F. (2009). An adaptive large neighborhood search algorithm for the resource-constrained project scheduling problem. In *Proceedings of the VIII Metaheuristics International Conference (MIC) 2009*. Hamburg, Germany.

Pisinger, D., & Røpke, S. (2007). A general heuristic for vehicle routing problems. *Computers and Operations Research*, *34*, 2403–2435.

Pisinger, D., & Røpke, S. (2010). Large neighborhood search. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics*. Springer Verlag. (2nd ed.).

Pochet, Y., & Wolsey, L. (2006). *Production Planning in Mixed Integer Programming*. Springer.

Røpke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, *40*, 455–472.

Sahling, F., Buschkühl, L., Tempelmeier, H., & Helber, S. (2009). Solving a multi-level capacitated lot sizing problem with multi-period setup carryover via a fix-and-optimize heuristic. *Computers and Operations Research*, *36*, 2546–2553.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. *CP-98 (Fourth International Conference on Principles and Practice of Constraint Programming)*, *1520*, 417–431.

Sural, H., Denizel, M., & Wassenhove, L. V. (2009). Lagrangean based heuristics for lot-sizing with setup times. *European Journal of Operational Research*, *194*, 51–63.

Trigeiro, W., Thomas, L., & McClain, J. (1989). Capacitated lot sizing with setup times. *Management Science*, *35*, 353–366.

Wolsey, L. (2002). Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation. *Management Science*, *48*, 1587–1602.

Instance	ALNS					MIP STD					MIP TP				
	LB	UB	UB*	gap	tb(s)	LB	UB	gap	tb(s)	tt(s)	LB	UB	gap	t(s)	tt(s)
G51-10	931.2	1049.0	1049	12.65	3.18	1049.0	1049	0.0	2.44	2.96	1049.0	1049	0.0	2.6	4.6
G51	1842.4	2151.0	2151	16.75	19.48	1984.1	2151	8.41	271.32	-	1989.9	2151	8.1	249.58	-
G52-10	585.1	676.0	676	15.54	1.10	676.0	676	0.0	0.11	0.54	676.0	676	0.0	0.35	1.44
G52	1287.1	1599.0	1599	24.23	2.88	1446.7	1599	10.53	0.84	-	1421.4	1599	12.49	1.58	-
G53-10	606.7	663.0	663	9.29	0.78	663.0	663	0.0	0.07	0.24	663.0	663	0.0	0.1	0.27
G53	1181.4	1483.4	1473	25.57	23.51	1473.0	1473	0.0	15.2	158.22	1473.0	1473	0.0	26.13	138.9
G54-10	243.0	363.0	363	49.36	1.73	363.0	363	0.0	0.07	0.1	363.0	363	0.0	0.13	0.14
G54	852.8	1050.0	1050	23.12	1.78	1050.0	1050	0.0	0.06	9.4	1050.0	1050	0.0	0.11	37.98
G55-10	1281.5	1383.6	1380	7.97	25.47	1380.0	1380	0.0	0.8	13.21	1380.0	1380	0.0	0.2	10.68
G55	2880.1	3097.0	3097	7.53	99.47	3029.1	3097	2.24	0.7	-	3001.0	3097	3.2	24.82	-
G56-10	665.3	770.0	770	15.74	1.45	770.0	770	0.0	2.97	90.95	761.3	770	1.15	13.58	-
G56	2475.0	2600.0	2600	5.05	54.43	2504.6	2600	3.81	51.88	-	2489.8	2600	4.43	58.08	-
G57-10	1701.4	1762.3	1758	3.58	66.12	1758.0	1758	0.0	8.63	25.12	1758.0	1758	0.0	2.5	8.29
G57	3270.8	3510.9	3502	7.34	126.70	3301.8	3504	6.12	273.77	-	3298.7	3504	6.22	272.34	-
G58-10	1942.9	2035.0	2035	4.74	58.30	2035.0	2035	0.0	19.78	158.68	2030.8	2035	0.21	19.02	-
G58	4153.1	4297.2	4293	3.47	96.89	4199.2	4295	2.28	93.19	-	4193.0	4293	2.38	292.48	-
G59-10	1911.2	1991.2	1990	4.19	47.84	1911.5	1990	4.1	276.42	-	1906.6	1990	4.38	1.34	-
G59	4639.2	4862.3	4852	4.81	165.83	4677.2	4863	3.97	283.84	-	4676.7	4887	4.5	297.94	-
G60-10	1439.1	1495.2	1490	3.9	60.72	1484.3	1490	0.39	169.77	-	1428.2	1494	4.6	271.34	-
G60	3709.6	4007.1	3992	8.02	107.08	3742.8	3979	6.31	287.28	-	3734.2	3985	6.72	287.01	-
G66-10	749.8	845.0	845	12.69	1.79	845.0	845	0.0	0.13	0.89	845.0	845	0.0	0.18	0.71
G66-15	1111.0	1346.8	1342	21.23	3.63	1342.0	1342	0.0	40.54	133.85	1328.3	1342	1.03	16.61	-
G66	3434.1	4351.6	4306	26.72	123.33	3606.0	4362	20.96	289.43	-	3589.2	4416	23.04	282.76	-
G67-10	447.5	480.0	480	7.26	0.38	480.0	480	0.0	0.05	0.07	480.0	480	0.0	0.1	0.24
G67-15	1567.5	1805.0	1805	15.15	27.43	1760.6	1805	2.52	215.56	-	1717.9	1805	5.07	71.89	-
G67	3752.5	4382.2	4378	16.78	105.09	3841.0	4433	15.41	281.27	-	3847.6	4429	15.11	280.32	-
G68-10	790.8	924.1	922	16.85	32.86	922.0	922	0.0	1.55	127.57	922.0	922	0.0	13.79	41.83
G68-15	2441.6	2653.8	2650	8.69	91.16	2514.7	2650	5.38	285.74	-	2528.7	2650	4.8	272.74	-
G68	7342.3	8121.7	8097	10.62	238.33	7341.8	8244	12.29	299.93	-	7319.3	8307	13.49	281.46	-
G69-10	118.0	186.0	186	57.64	0.04	186.0	186	0.0	0.03	0.05	186.0	186	0.0	0.07	0.07
G69-15	600.0	827.0	827	37.84	2.86	827.0	827	0.0	0.62	136.25	827.0	827	0.0	1.44	95.33
G69	2271.8	2919.0	2919	28.49	96.93	2369.8	2950	24.49	299.79	-	2399.5	2950	22.94	276.49	-
G70-10	694.0	795.0	795	14.56	1.15	795.0	795	0.0	0.13	5.85	795.0	795	0.0	0.13	5.18
G70-15	1558.2	1752.0	1752	12.44	14.84	1721.6	1752	1.77	9.82	-	1728.0	1752	1.39	14.3	-
G70	3933.8	4763.5	4754	21.09	204.25	4060.7	4770	17.47	289.39	-	4031.9	4791	18.83	286.66	-
G71-10	632.6	701.0	700.998	10.81	25.98	701.0	701	0.0	0.29	2.32	701.0	701	0.0	2.08	12.98
G71-15	708.5	891.0	891	25.76	14.37	891.0	891	0.0	1.74	114.52	891.0	891	0.0	33.22	113.67
G71	2019.6	2621.6	2620	29.81	128.09	1959.0	2634	34.46	296.21	-	2085.8	2675	28.25	295.97	-
G72-10	264.5	367.0	367	38.75	1.14	367.0	367	0.0	0.46	0.51	367.0	367	0.0	0.6	1.77
G72-15	512.3	711.0	711	38.79	11.94	221.0	711	221.67	274.34	-	650.3	711	9.33	270.5	-
G72	1059.7	1619.0	1619	52.78	59.98	273.1	1619	492.88	297.3	-	1196.0	1633	36.54	294.73	-
G73-10	1637.1	1772.0	1772	8.24	70.35	1721.5	1772	2.93	7.08	-	1692.6	1772	4.69	71.49	-
G73-15	3645.0	3850.9	3841	5.65	123.21	3637.7	3871	6.41	276.34	-	3633.0	3841	5.73	284.46	-
G73	9451.0	10143.0	10100	7.32	251.91	9472.9	10175	7.41	299.83	-	9468.8	10271	8.47	274.46	-
G74-10	780.2	860.0	860	10.23	6.52	860.0	860	0.0	44.75	126.79	860.0	860	0.0	27.96	147.44
G74-15	1857.1	2182.2	2142	17.51	50.28	1947.8	2234	14.69	2.58	-	1926.7	2234	15.95	293.4	-
G74	3960.2	4688.3	4663	18.39	205.13	3987.5	4712	18.17	297.55	-	3998.6	4739	18.52	300.28	-
G75-10	1457.7	1628.0	1628	11.68	8.30	1514.4	1628	7.5	270.19	-	1513.7	1628	7.55	0.69	-
G75-15	3446.0	3657.0	3656	6.12	91.05	3448.3	3656	6.02	281.33	-	3444.9	3668	6.48	284.86	-
G75	9987.0	10592.8	10545	6.07	265.03	10005.1	10724	7.19	297.95	-	10005.6	10826	8.2	278.25	-
50	13.42	0.18	0.01	16.98	64.44	7.17	0.3	19.36	128.42	196.17	5.0	0.46	6.28	126.66	204.68

Table A.3: Detailed results for the homogeneous class S instances. Columns are similar to those of Table 1 with the addition of tt(s) that is the total time in seconds used to prove optimality (a '-' indicates that the instance was not proved optimal). Note that for the MIP solver, some solution times may be above the 300 seconds time limit due to close-down time of ILOG CPLEX.

Instance	ALNS					MIP STD					MIP TP				
	LB	UB	UB*	gap	tb(s)	LB	UB	gap	tb(s)	tt(s)	LB	UB	gap	t(s)	tt(s)
G51-10	1380.8	1441.0	1441	4.36	0.89	1441.0	1441	0.0	0.11	0.11	1441.0	1441	0.0	0.12	0.13
G51	3579.3	3684.0	3684	2.93	8.56	3684.0	3684	0.0	0.44	1.09	3684.0	3684	0.0	0.54	2.4
G52-10	664.8	761.1	761	14.48	42.46	761.0	761	0.0	0.08	0.14	761.0	761	0.0	0.12	0.24
G52	1470.3	1773.0	1773	20.59	3.04	1773.0	1773	0.0	0.56	2.66	1773.0	1773	0.0	6.61	8.85
G53-10	794.6	842.0	842	5.97	0.28	842.0	842	0.0	0.06	0.08	842.0	842	0.0	0.03	0.07
G53	1894.3	2169.0	2169	14.5	32.89	2169.0	2169	0.0	6.09	9.58	2169.0	2169	0.0	2.46	6.68
G54-10	119.2	424.0	424	255.67	0.01	424.0	424	0.0	0.02	0.02	424.0	424	0.0	0.02	0.03
G54	2123.0	2183.0	2183	2.82	0.14	2183.0	2183	0.0	0.19	0.2	2183.0	2183	0.0	0.21	0.27
G55-10	1842.3	1940.0	1940	5.3	5.50	1940.0	1940	0.0	0.18	0.19	1940.0	1940	0.0	0.29	0.29
G55	4981.6	5298.4	5290	6.36	45.32	5290.0	5290	0.0	19.61	21.93	5290.0	5290	0.0	8.75	18.78
G56-10	1127.7	1183.0	1183	4.91	1.33	1183.0	1183	0.0	0.1	0.19	1183.0	1183	0.0	0.19	0.27
G56	5318.9	5586.7	5585	5.03	37.82	5585.0	5585	0.0	31.14	32.85	5585.0	5585	0.0	49.43	52.35
G57-10	2023.2	2124.0	2124	4.98	1.16	2124.0	2124	0.0	0.34	1.86	2124.0	2124	0.0	0.57	5.02
G57	4245.1	4585.3	4583	8.01	72.70	4479.3	4576	2.16	249.61	-	4462.0	4576	2.55	100.01	-
G58-10	2863.2	2902.0	2902	1.35	13.06	2902.0	2902	0.0	1.42	1.63	2902.0	2902	0.0	1.54	1.7
G58	7082.1	7320.7	7319	3.37	41.66	7251.5	7318	0.92	47.75	-	7276.2	7318	0.57	117.37	-
G59-10	3189.2	3307.7	3306	3.72	72.43	3306.0	3306	0.0	2.91	7.39	3306.0	3306	0.0	5.44	6.88
G59	9687.6	9965.1	9942	2.86	163.73	9941.0	9942	0.01	225.8	226.94	9941.0	9942	0.01	69.49	287.64
G60-10	2619.2	2737.0	2737	4.5	7.16	2737.0	2737	0.0	0.99	14.42	2737.0	2737	0.0	6.01	21.08
G60	8292.3	8492.2	8492	2.41	116.46	8428.7	8492	0.75	226.52	-	8366.0	8494	1.53	115.44	-
G66-10	974.7	1063.0	1063	9.05	0.13	1063.0	1063	0.0	0.06	0.15	1063.0	1063	0.0	0.13	0.76
G66-15	1545.4	1733.0	1733	12.14	3.93	1733.0	1733	0.0	0.3	4.74	1733.0	1733	0.0	0.34	5.02
G66	5461.1	6213.0	6203	13.77	93.06	5679.1	6191	9.01	279.67	-	5671.4	6203	9.37	288.31	-
G67-10	486.0	486.0	486	0.0	0.03	486.0	486	0.0	0.01	0.03	486.0	486	0.0	0.01	0.02
G67-15	2539.7	2929.0	2929	15.33	6.82	2929.0	2929	0.0	0.39	0.68	2929.0	2929	0.0	0.62	1.48
G67	7080.9	8610.9	8533	21.61	179.80	7777.7	8539	9.79	289.45	-	7526.0	8533	13.38	283.78	-
G68-10	1359.1	1534.0	1534	12.87	2.43	1534.0	1534	0.0	0.5	0.65	1534.0	1534	0.0	0.15	1.24
G68-15	4975.4	5410.9	5401	8.75	45.70	5287.2	5401	2.15	47.18	-	5157.1	5401	4.73	53.63	-
G68	16261.0	17835.1	17745	9.68	208.52	16287.7	17951	10.21	276.06	-	16247.8	17874	10.01	284.76	-
G69-10	37.5	189.0	189	404.4	0.01	189.0	189	0.0	0.03	0.03	189.0	189	0.0	0.01	0.03
G69-15	729.3	971.0	971	33.14	0.63	971.0	971	0.0	0.49	1.3	971.0	971	0.0	0.28	1.17
G69	3217.1	4124.6	4093	28.21	121.06	3587.9	4129	15.08	290.33	-	3620.8	4179	15.42	298.53	-
G70-10	1884.9	2021.0	2021	7.22	0.25	2021.0	2021	0.0	0.06	0.1	2021.0	2021	0.0	0.11	0.17
G70-15	4928.3	5397.0	5397	9.51	62.72	5397.0	5397	0.0	0.75	60.44	5397.0	5397	0.0	0.45	154.19
G70	12496.2	14419.2	14366	15.39	203.84	12876.7	14515	12.72	278.49	-	12873.8	14462	12.34	286.87	-
G71-10	770.8	815.0	815	5.74	26.11	815.0	815	0.0	0.06	0.08	815.0	815	0.0	0.09	0.13
G71-15	884.0	1056.0	1056	19.46	0.24	1056.0	1056	0.0	0.06	0.49	1056.0	1056	0.0	0.7	1.26
G71	2879.4	3819.4	3803	32.65	162.19	3064.0	3856	25.85	278.81	-	3227.2	3828	18.62	293.67	-
G72-10	330.6	376.0	376	13.72	0.76	376.0	376	0.0	0.21	0.24	376.0	376	0.0	0.25	0.28
G72-15	546.6	743.0	743	35.93	5.99	331.2	743	124.32	240.63	-	743.0	743	0.0	5.72	7.85
G72	1212.6	1724.0	1724	42.18	64.54	302.4	1724	470.19	298.29	-	1377.3	1724	25.18	278.58	-
G73-10	2636.4	2681.0	2681	1.69	2.77	2681.0	2681	0.0	0.26	0.38	2681.0	2681	0.0	0.25	0.47
G73-15	6730.5	6935.0	6935	3.04	67.63	6803.8	6935	1.93	271.95	-	6807.8	6935	1.87	271.02	-
G73	19527.5	20471.1	20385	4.83	284.89	19542.4	20701	5.93	288.96	-	19559.5	20400	4.3	286.97	-
G74-10	905.7	988.0	988	9.08	7.82	988.0	988	0.0	3.21	4.04	988.0	988	0.0	0.32	3.09
G74-15	2335.4	2613.7	2610	11.92	42.86	2610.0	2610	0.0	8.74	20.8	2610.0	2610	0.0	35.92	47.55
G74	4989.5	5873.4	5857	17.71	218.91	5103.8	6024	18.03	289.72	-	5076.8	5974	17.67	292.22	-
G75-10	2145.9	2227.0	2227	3.78	1.38	2227.0	2227	0.0	0.17	0.94	2227.0	2227	0.0	1.34	1.91
G75-15	6001.6	6204.4	6198	3.38	128.88	6175.1	6196	0.34	270.52	-	6167.1	6203	0.58	288.41	-
G75	20618.7	21702.5	21511	5.26	287.26	20742.1	21537	3.83	296.15	-	20778.8	21433	3.15	295.47	-
50	11.8	0.13	0.02	23.71	57.96	4.67	0.19	14.26	90.51	110.34	2.35	0.13	2.83	80.67	109.0

Table A.4: Detailed results for the heterogeneous class S instances. Columns are similar to those of Table 1 with the addition of tt(s) that is the total time in seconds used to prove optimality (a '-' indicates that the instance was not proved optimal). Note that for the MIP solver, some solution times may be above the 300 seconds time limit due to close-down time of ILOG CPLEX.

Instance	ALNS					MIP STD					MIP TP				
	LB	UB	UB*	gap	tb(s)	LB	UB	gap	tb(s)	tt(s)	LB	UB	gap	t(s)	tt(s)
G51-30	3633.4	4303.6	4302	18.45	112.75	3734.1	4302	15.21	278.51	-	3713.6	4342	16.92	283.69	-
G51-45	5422.0	6484.3	6453	19.59	271.69	5485.9	6517	18.8	293.86	-	5463.3	6549	19.87	299.78	-
G52-30	2514.7	3198.0	3198	27.17	21.81	2607.5	3198	22.65	102.17	-	2598.5	3198	23.07	282.91	-
G52-45	3734.3	4797.0	4797	28.46	101.89	3798.3	4797	26.29	281.83	-	3763.5	4819	28.05	280.51	-
G53-30	2382.1	2968.1	2959	24.6	72.85	2547.5	2972	16.66	7.11	-	2533.0	2972	17.33	90.92	-
G53-45	3549.5	4456.7	4445	25.56	99.35	3667.8	4458	21.54	179.51	-	3676.2	4458	21.27	285.02	-
G54-30	1632.6	2100.0	2100	28.63	11.86	1870.4	2100	12.27	0.19	-	1844.8	2100	13.84	9.97	-
G54-45	2388.5	3150.0	3150	31.88	9.10	2589.7	3150	21.64	0.48	-	2598.7	3150	21.21	7.51	-
G55-30	5723.2	6227.8	6194	8.82	180.04	5748.8	6252	8.75	275.42	-	5739.5	6275	9.33	289.45	-
G55-45	8553.1	9369.2	9333	9.54	232.26	8550.1	9399	9.93	274.27	-	8541.8	9441	10.53	290.62	-
G56-30	4894.5	5218.3	5200	6.62	174.58	4879.0	5207	6.72	296.73	-	4869.0	5285	8.54	282.92	-
G56-45	7324.8	7870.3	7822	7.45	264.74	7281.3	7829	7.52	292.23	-	7275.1	8043	10.56	303.68	-
G57-30	6508.3	7054.6	7032	8.39	238.69	6521.5	7006	7.43	291.45	-	6519.5	7078	8.57	298.85	-
G57-45	9756.4	10654.5	10588	9.21	291.79	9749.1	10584	8.56	291.83	-	9743.7	10804	10.88	288.81	-
G58-30	8306.0	8609.1	8594	3.65	195.35	8326.8	8632	3.67	294.53	-	8326.7	8637	3.73	279.23	-
G58-45	12452.2	12984.1	12938	4.27	277.81	12466.8	13007	4.33	279.07	-	12466.4	13107	5.14	301.21	-
G59-30	9271.4	9807.6	9757	5.78	279.66	9281.2	9865	6.29	290.74	-	9268.6	9886	6.66	158.31	-
G59-45	13908.4	14902.4	14836	7.15	288.31	13892.0	14855	6.93	293.45	-	13880.7	14993	8.01	291.66	-
G60-30	7399.9	8017.0	7989	8.34	221.78	7409.0	8070	8.92	297.61	-	7395.6	8193	10.78	286.69	-
G60-45	11094.1	12097.0	12065	9.04	291.33	11083.8	12312	11.08	273.7	-	11081.7	12311	11.09	300.74	-
G66-60	6870.8	8804.7	8741	28.15	255.16	7002.8	8842	26.26	298.03	-	6964.1	8949	28.5	295.99	-
G66-90	10314.6	13347.4	13236	29.4	266.60	10367.2	13468	29.91	298.68	-	10356.8	13782	33.07	326.51	-
G67-60	7497.2	8862.4	8812	18.21	285.74	7569.0	8990	18.77	296.68	-	7530.9	9050	20.17	294.93	-
G67-90	11242.4	13570.5	13463	20.71	285.79	11275.6	13419	19.01	299.91	-	11219.8	13752	22.57	313.8	-
G68-60	14633.3	16700.6	16599	14.13	269.84	14600.8	16574	13.51	299.77	-	14591.0	16871	15.63	302.43	-
G68-90	21923.2	25390.1	25247	15.81	254.20	21868.4	25226	15.35	296.7	-	21857.1	25789	17.99	325.6	-
G69-60	4453.0	5862.7	5838	31.66	265.06	4534.5	5939	30.97	292.4	-	4520.5	6204	37.24	302.39	-
G69-90	6682.9	8906.8	8855	33.28	286.74	6701.6	9090	35.64	299.44	-	6699.3	9343	39.46	325.03	-
G70-60	7811.7	9674.0	9596	23.84	278.81	7887.8	9718	23.2	297.66	-	7889.0	9913	25.66	304.77	-
G70-90	11738.9	15158.0	14914	29.13	285.56	11748.7	14804	26.01	299.76	-	11717.5	15294	30.52	324.97	-
G71-60	3941.9	5325.2	5250	35.09	283.49	633.5	5418	755.29	299.88	-	4054.2	5432	33.99	317.89	-
G71-90	5923.2	8267.8	8142	39.58	284.12	613.5	9016	1369.5	299.89	-	5965.5	8239	38.11	402.38	-
G72-60	2107.7	3324.8	3294	57.74	272.93	235.5	3617	1435.89	300.02	-	2043.5	3458	69.22	317.5	-
G72-90	3183.4	5160.9	5063	62.12	283.59	216.1	5279	2342.74	299.98	-	2933.3	5108	74.14	413.22	-
G73-60	18900.5	20715.2	20639	9.6	286.50	18918.1	20775	9.82	298.73	-	18906.0	20618	9.06	321.61	-
G73-90	28351.1	31206.7	31049	10.07	271.96	28356.8	30860	8.83	298.45	-	28348.9	30987	9.31	419.86	-
G74-60	7906.6	9684.7	9516	22.49	284.24	7924.6	9792	23.57	298.37	-	7937.5	9778	23.19	320.85	-
G74-90	11875.9	14843.2	14716	24.99	285.09	11866.1	14635	23.33	299.21	-	11850.5	15016	26.71	416.42	-
G75-60	19974.5	21615.9	21467	8.22	281.03	19977.7	21614	8.19	298.95	-	19966.1	21946	9.92	319.36	-
G75-90	29959.9	32737.0	32644	9.27	275.89	29948.8	32666	9.07	300.02	-	29933.6	32873	9.82	391.57	-
40	15.44	0.67	0.07	20.4	229.5	20.56	1.3	161.75	264.18	300.08	14.73	1.89	20.99	291.74	323.52

Table A.5: Detailed results for the homogeneous class M instances. Columns are similar to those of Table 1 with the addition of tt(s) that is the total time in seconds used to prove optimality (a '-' indicates that the instance was not proved optimal). Note that for the MIP solver, some solution times may be above the 300 seconds time limit due to close-down time of ILOG CPLEX.

Instance	ALNS					MIP STD					MIP TP				
	LB	UB	UB*	gap	tb(s)	LB	UB	gap	tb(s)	tt(s)	LB	UB	gap	t(s)	tt(s)
G51-30	7103.5	7368.0	7368	3.72	66.40	7346.7	7368	0.29	203.21	-	7231.7	7368	1.88	282.81	-
G51-45	10635.0	11052.0	11052	3.92	158.16	10645.0	11052	3.82	276.38	-	10509.9	11136	5.96	255.15	-
G52-30	2943.7	3546.0	3546	20.46	19.63	3255.3	3582	10.04	270.06	-	3212.5	3546	10.38	10.26	-
G52-45	4393.5	5344.2	5319	21.64	81.09	4602.9	5368	16.62	274.87	-	4507.4	5363	18.98	289.25	-
G53-30	3787.4	4350.9	4338	14.88	159.36	4008.3	4380	9.27	296.63	-	4001.3	4380	9.46	219.01	-
G53-45	5678.9	6557.9	6507	15.48	239.63	5810.7	6574	13.14	298.56	-	5798.1	6688	15.35	275.32	-
G54-30	4222.0	4366.0	4366	3.41	15.65	4366.0	4366	0.0	0.31	2.37	4366.0	4366	0.0	2.81	7.09
G54-45	6302.4	6549.0	6549	3.91	34.81	6548.0	6549	0.02	2.77	100.1	6513.3	6549	0.55	11.57	-
G55-30	9893.0	10640.3	10580	7.55	233.74	10101.7	10593	4.86	276.46	-	10123.1	10785	6.54	287.4	-
G55-45	14783.0	16170.8	15935	9.39	272.15	14889.5	16070	7.93	290.26	-	14858.0	15990	7.62	281.43	-
G56-30	10568.8	11277.5	11213	6.71	193.42	10821.9	11237	3.84	279.92	-	10704.0	11348	6.02	276.34	-
G56-45	15797.3	16938.6	16858	7.22	261.75	15914.1	16929	6.38	289.14	-	15817.9	16943	7.11	294.74	-
G57-30	8447.5	9217.1	9182	9.11	195.16	8460.1	9182	8.53	44.3	-	8445.1	9352	10.74	5.16	-
G57-45	12628.0	13853.6	13795	9.71	198.80	12623.6	14238	12.79	2.24	-	12591.6	13908	10.45	39.16	-
G58-30	14149.3	14681.4	14638	3.76	230.47	14243.2	14685	3.1	270.8	-	14285.6	14695	2.87	16.63	-
G58-45	21211.7	22161.6	22055	4.48	283.90	21232.6	22081	4.0	275.67	-	21299.5	22042	3.49	306.47	-
G59-30	19337.8	20281.7	20236	4.88	280.35	19423.8	19947	2.69	299.66	-	19499.8	20098	3.07	295.8	-
G59-45	28997.8	30909.8	30637	6.59	276.55	29061.1	30563	5.17	296.37	-	29138.9	30773	5.61	299.09	-
G60-30	16571.2	17044.6	16987	2.86	288.14	16604.5	17073	2.82	274.85	-	16570.5	17104	3.22	275.48	-
G60-45	24849.9	25931.2	25791	4.35	280.39	24869.1	25638	3.09	294.97	-	24817.0	25829	4.08	298.12	-
G66-60	10913.7	12671.5	12464	16.11	257.85	11039.8	12642	14.51	291.15	-	11039.4	12812	16.06	179.34	-
G66-90	16355.9	19260.7	18901	17.76	286.09	16443.4	18985	15.46	296.29	-	16414.6	19879	21.11	320.11	-
G67-60	13951.8	18038.9	17656	29.29	291.98	14300.4	17399	21.67	299.84	-	13717.3	17968	30.99	305.69	-
G67-90	20797.0	27224.3	26982	30.9	280.03	20825.0	26537	27.43	299.9	-	20045.8	26741	33.4	314.78	-
G68-60	32375.6	36315.6	36088	12.17	279.31	32292.0	35992	11.46	295.88	-	32255.2	35891	11.27	304.37	-
G68-90	48475.4	54950.7	54481	13.36	278.92	48361.1	54223	12.12	295.97	-	48293.9	54945	13.77	324.74	-
G69-60	6423.7	8302.3	8269	29.24	227.37	6664.7	8401	26.05	289.79	-	6629.4	8552	29.0	295.43	-
G69-90	9654.3	12577.3	12459	30.28	273.90	9782.8	12748	30.31	297.31	-	9718.0	13275	36.6	320.7	-
G70-60	24925.5	29187.2	28955	17.1	284.67	25239.1	29375	16.39	291.95	-	25153.5	29697	18.06	302.93	-
G70-90	37389.5	45072.3	44772	20.55	287.02	37632.0	43981	16.87	295.89	-	37407.5	45021	20.35	327.5	-
G71-60	5808.3	7769.1	7690	33.76	289.09	774.4	8061	940.9	294.17	-	5988.6	8046	34.36	308.31	-
G71-90	8740.7	12075.9	11824	38.16	284.05	856.1	13082	1428.08	299.45	-	8730.4	12116	38.78	416.1	-
G72-60	2484.1	3501.3	3454	40.95	278.58	330.4	3627	997.84	296.91	-	2380.5	3679	54.55	317.76	-
G72-90	3667.3	5565.1	5427	51.75	286.58	291.1	5779	1885.34	299.84	-	3453.9	5486	58.83	418.55	-
G73-60	38992.7	41590.9	41352	6.66	273.98	38985.0	41103	5.43	295.73	-	39011.1	41352	6.0	320.25	-
G73-90	58468.4	63192.7	62987	8.08	258.80	58423.6	62408	6.82	298.88	-	58477.4	62803	7.4	271.69	-
G74-60	9985.0	12115.1	11909	21.33	287.41	10023.9	12324	22.95	296.89	-	10018.5	12178	21.56	303.02	-
G74-90	14995.2	18473.5	18241	23.2	283.09	15005.3	18142	20.9	291.21	-	14981.1	18848	25.81	409.23	-
G75-60	41221.6	44762.1	43794	8.59	276.80	41308.6	43579	5.5	298.16	-	41296.6	43875	6.24	318.22	-
G75-90	61823.9	67228.9	66664	8.74	265.99	61833.2	66533	7.6	300.11	-	61829.8	66480	7.52	399.69	-
40	11.68	1.11	0.28	15.55	232.53	16.92	1.19	140.8	261.07	287.63	11.03	1.66	15.63	262.51	316.3

Table A.6: Detailed results for the heterogeneous class M instances. Columns are similar to those of Table 1 with the addition of tt(s) that is the total time in seconds used to prove optimality (a '-' indicates that the instance was not proved optimal). Note that for the MIP solver, some solution times may be above the 300 seconds time limit due to close-down time of ILOG CPLEX.